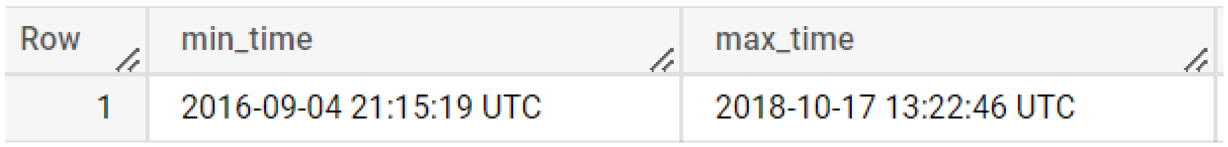
1. Initial exploration of dataset:
   1. Data type of columns:
      Numeric, Date and Time, Strings

   2. Time period for which the data is given: 2016-09 to 2018-10
      i. Query:

```
SELECT MIN(order_purchase_timestamp) AS min_time,
MAX(order_delivered_customer_date) AS max_time
FROM Target.orders
```

      ii. Output:

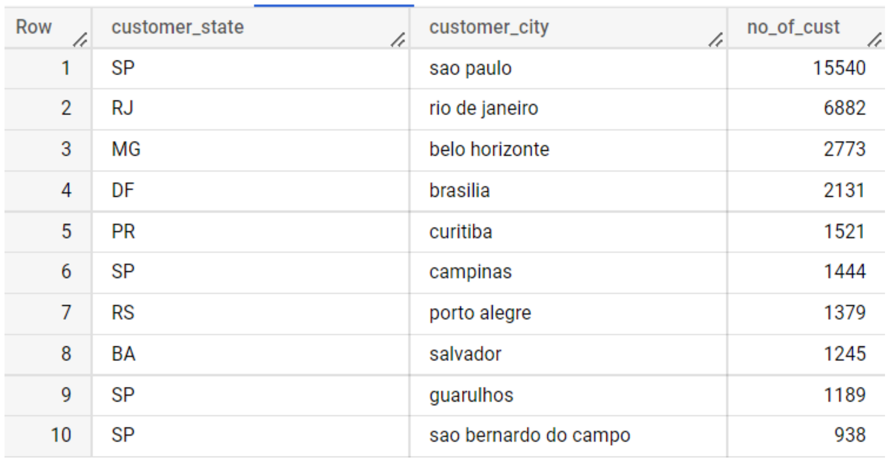


Row	min_time	max_time
1	2016-09-04 21:15:19 UTC	2018-10-17 13:22:46 UTC

   3. Cities and states covered
      i. Query:

```
SELECT customer_state, customer_city, COUNT(*) AS no_of_cust
FROM Target.customers
GROUP BY customer_state, customer_city
ORDER BY no_of_cust DESC
```

      ii. Output:



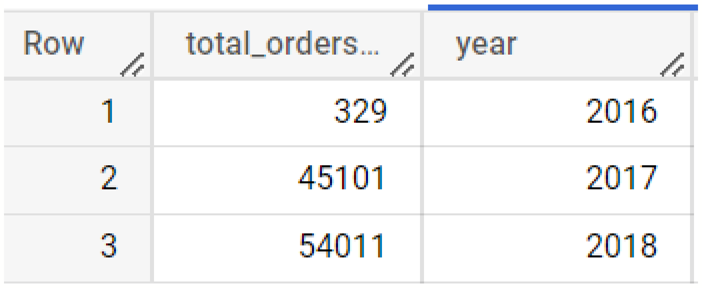
Row	customer_state	customer_city	no_of_cust
1	SP	sao paulo	15540
2	RJ	rio de janeiro	6882
3	MG	belo horizonte	2773
4	DF	brasilia	2131
5	PR	curitiba	1521
6	SP	campinas	1444
7	RS	porto alegre	1379
8	BA	salvador	1245
9	SP	guarulhos	1189
10	SP	sao bernardo do campo	938

2. In - depth Exploration:
   1. There is a whooping rise of people buying from e-commerce websites in Brazil. From 329 orders in 2016 to 54011 orders in 2018.
      i. Query:

```
SELECT COUNT(order_id) AS total_orders_placed,
EXTRACT(YEAR FROM order_purchase_timestamp) AS year
FROM Target.orders
GROUP BY year
```

```
ORDER BY year
```

ii. Output:

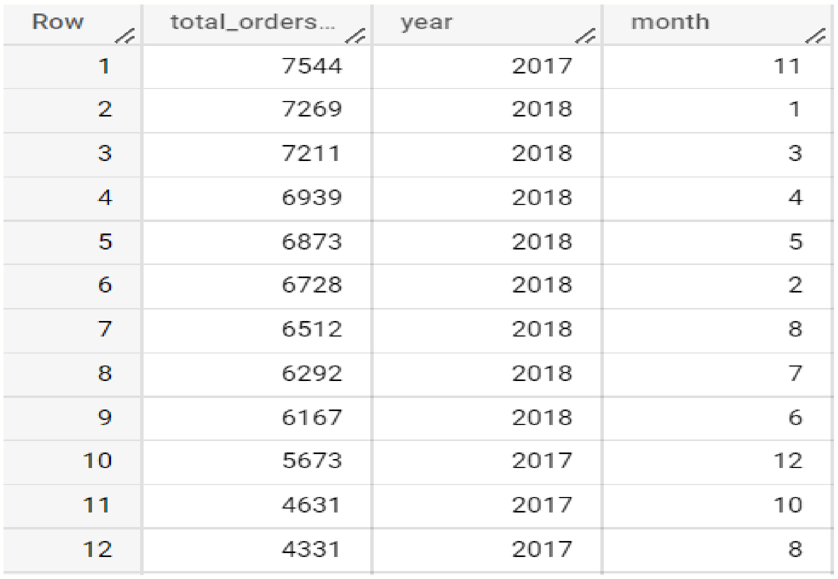


Row	total_orders...	year
1	329	2016
2	45101	2017
3	54011	2018

i. Query:

```
SELECT COUNT(order_id) AS total_orders_placed,
EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS month
FROM Target.orders
GROUP BY year, month
ORDER BY total_orders_placed DESC
```

ii. Output:



Row	total_orders...	year	month
1	7544	2017	11
2	7269	2018	1
3	7211	2018	3
4	6939	2018	4
5	6873	2018	5
6	6728	2018	2
7	6512	2018	8
8	6292	2018	7
9	6167	2018	6
10	5673	2017	12
11	4631	2017	10
12	4331	2017	8

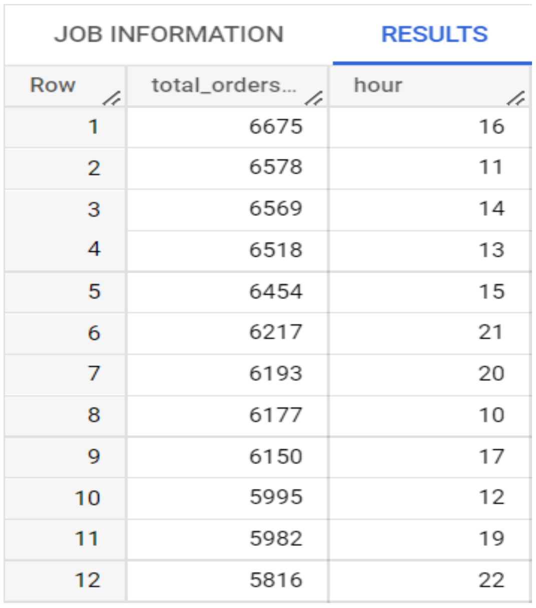
2. The maximum orders are received during the night time i.e. from 18 - 1. The least orders are received during dawn.

i. Query:

```
SELECT COUNT(order_id) AS total_orders_placed,
  EXTRACT(HOUR FROM order_purchase_timestamp) AS hour
FROM Target.orders
```

```
GROUP BY hour
ORDER BY total_orders_placed DESC
```

ii. Output:



JOB INFORMATION RESULTS

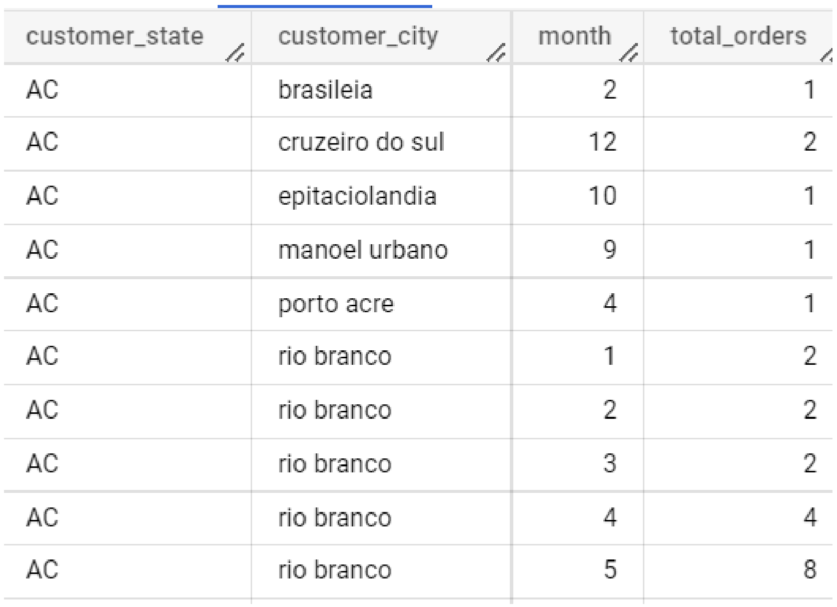
Row	total_orders...	hour
1	6675	16
2	6578	11
3	6569	14
4	6518	13
5	6454	15
6	6217	21
7	6193	20
8	6177	10
9	6150	17
10	5995	12
11	5982	19
12	5816	22

3. Evolution of E-commerce orders in the Brazil region:
    1. Get month on month orders by region, states:

        i. Query:

```
SELECT c.customer_state, c.customer_city,
EXTRACT(MONTH FROM o.order_purchase_timestamp) AS month,
COUNT(o.order_id) AS total_orders
FROM Target.orders o JOIN Target.customers c
ON o.customer_id = c.customer_id
WHERE EXTRACT(YEAR FROM o.order_purchase_timestamp) = 2017
GROUP BY c.customer_state, c.customer_city, month
ORDER BY c.customer_state, c.customer_city, month
```

        ii. Output:



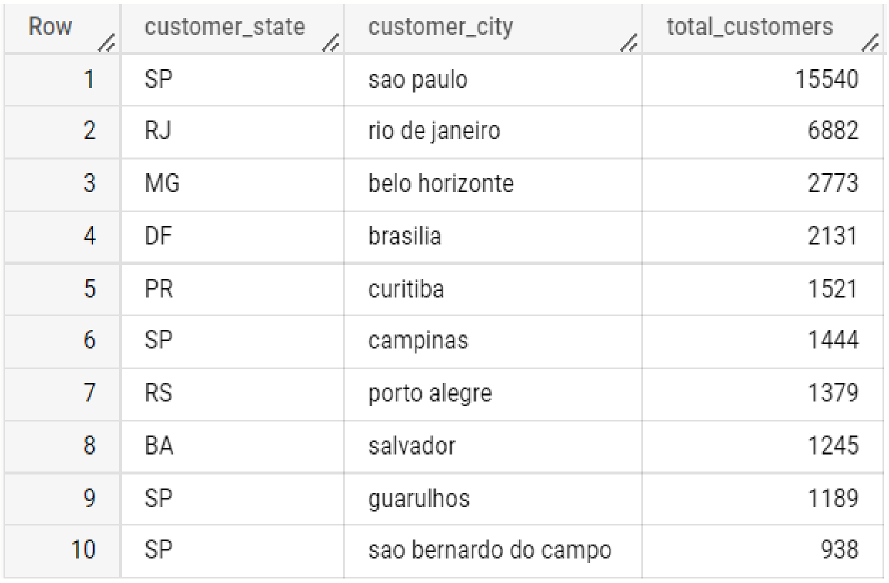
customer_state	customer_city	month	total_orders
AC	brasileia	2	1
AC	cruzeiro do sul	12	2
AC	epitaciolandia	10	1
AC	manoel urbano	9	1
AC	porto acre	4	1
AC	rio branco	1	2
AC	rio branco	2	2
AC	rio branco	3	2
AC	rio branco	4	4
AC	rio branco	5	8

2. Customer distribution in Brazil: The maximum number of orders are received for the 3 states which is SP, RJ, MG

i. Query:

```
SELECT customer_state, customer_city,
COUNT(customer_id) AS total_customers
FROM Target.customers
GROUP BY customer_state, customer_city
ORDER BY total_customers DESC
```

ii. Output:



Row	customer_state	customer_city	total_customers
1	SP	sao paulo	15540
2	RJ	rio de janeiro	6882
3	MG	belo horizonte	2773
4	DF	brasilia	2131
5	PR	curitiba	1521
6	SP	campinas	1444
7	RS	porto alegre	1379
8	BA	salvador	1245
9	SP	guarulhos	1189
10	SP	sao bernardo do campo	938

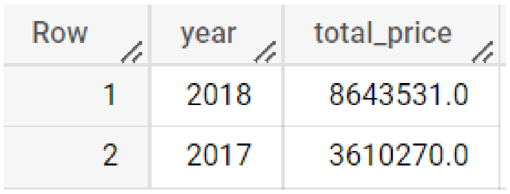
4. Impact on Economy:

1. Get % increase in cost of orders from 2017 to 2018: There is a whopping rise of 139% in the cost of orders from 2017 to 2018.

i. Query:

```
SELECT EXTRACT(YEAR FROM o.order_purchase_timestamp) AS year,
ROUND(SUM(i.price + i.freight_value)) AS total_price
FROM Target.orders o JOIN Target.order_items i
ON o.order_id = i.order_id
WHERE EXTRACT(MONTH FROM o.order_purchase_timestamp) BETWEEN 1 AND
8
GROUP BY year
```

ii. Output:



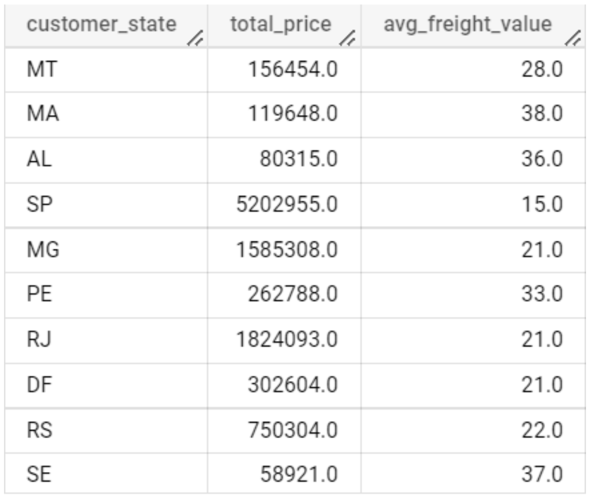
Row	year	total_price
1	2018	8643531.0
2	2017	3610270.0

2. Mean & Sum of price and freight value by customer state

i. Query:

```
SELECT c.customer_state,
ROUND(SUM(i.price)) AS total_price,
ROUND(AVG(i.freight_value)) AS avg_freight_value
FROM Target.customers c JOIN Target.orders o
ON c.customer_id = o.customer_id
JOIN Target.order_items i
ON o.order_id = i.order_id
GROUP BY c.customer_state
```

ii. Output:

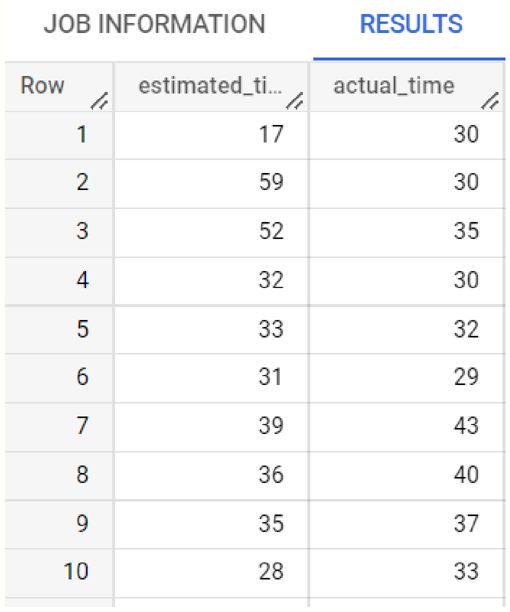


customer_state	total_price	avg_freight_value
MT	156454.0	28.0
MA	119648.0	38.0
AL	80315.0	36.0
SP	5202955.0	15.0
MG	1585308.0	21.0
PE	262788.0	33.0
RJ	1824093.0	21.0
DF	302604.0	21.0
RS	750304.0	22.0
SE	58921.0	37.0

5. Analysis on sale, freight and delivery time:
   1. Days between purchasing, delivering and estimated delivery
      i. Query:

```
SELECT DATE_DIFF( order_estimated_delivery_date,
order_purchase_timestamp, DAY) AS estimated_time,
DATE_DIFF(order_delivered_customer_date, order_purchase_timestamp,
DAY) AS actual_time
FROM Target.orders
WHERE DATE_DIFF(order_delivered_customer_date,
order_purchase_timestamp, DAY) IS NOT NULL
```

      ii. Output:



JOB INFORMATION RESULTS

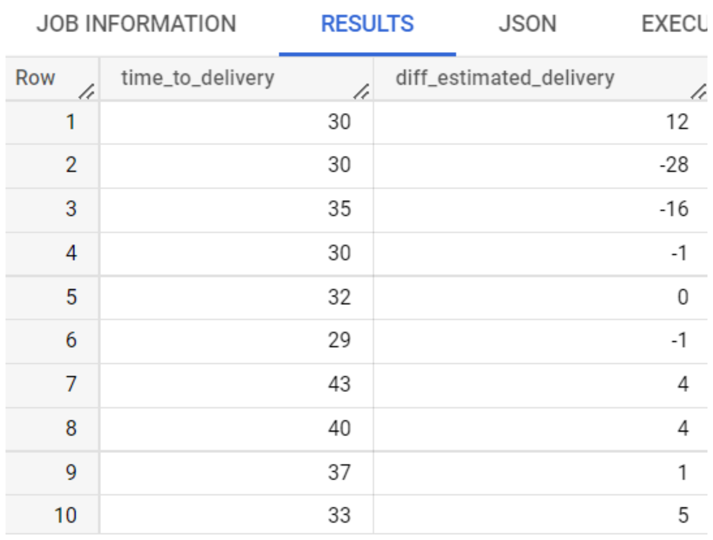
Row	estimated_ti...	actual_time
1	17	30
2	59	30
3	52	35
4	32	30
5	33	32
6	31	29
7	39	43
8	36	40
9	35	37
10	28	33

2. Create columns:

i. Query:

```
SELECT DATE_DIFF(order_purchase_timestamp,
order_delivered_customer_date, DAY) AS time_to_delivery,
DATE_DIFF(order_estimated_delivery_date,
order_delivered_customer_date, DAY) AS diff_estimated_delivery
FROM Target.orders
```

ii. Output:



Row	time_to_delivery	diff_estimated_delivery
1	30	12
2	30	-28
3	35	-16
4	30	-1
5	32	0
6	29	-1
7	43	4
8	40	4
9	37	1
10	33	5

3. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

i. Query:

```
SELECT c.customer_state,
ROUND(AVG(i.freight_value)) AS mean_freight_value,
ROUND(AVG(DATE_DIFF(order_purchase_timestamp,
order_delivered_customer_date, DAY))) AS time_to_delivery,
ROUND(AVG(DATE_DIFF(order_estimated_delivery_date,
order_delivered_customer_date, DAY))) AS diff_estimated_delivery
FROM Target.orders o JOIN Target.customers c
ON o.customer_id = c.customer_id
JOIN Target.order_items i
ON o.order_id = i.order_id
GROUP BY c.customer_state
```

ii. Output:

customer_state	mean_freight_value	time_to_delivery	diff_estimated_delivery
MT	28.0	-18.0	14.0
MA	38.0	-21.0	9.0
AL	36.0	-24.0	8.0
SP	15.0	-8.0	10.0
MG	21.0	-12.0	12.0
PE	33.0	-18.0	13.0
RJ	21.0	-15.0	11.0
DF	21.0	-13.0	11.0
RS	22.0	-15.0	13.0
SE	37.0	-21.0	9.0

4. Sort the data to get the following:

1. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

i. Query:

```
SELECT c.customer_state,
ROUND(AVG(i.freight_value)) AS mean_freight_value,
ROUND(AVG(DATE_DIFF(order_purchase_timestamp,
order_delivered_customer_date, DAY))) AS time_to_delivery,
ROUND(AVG(DATE_DIFF(order_estimated_delivery_date,
order_delivered_customer_date, DAY))) AS diff_estimated_delivery
FROM Target.orders o JOIN Target.customers c
ON o.customer_id = c.customer_id
JOIN Target.order_items i
ON o.order_id = i.order_id
GROUP BY c.customer_state
ORDER BY mean_freight_value DESC
LIMIT 5
```

ii. Output:

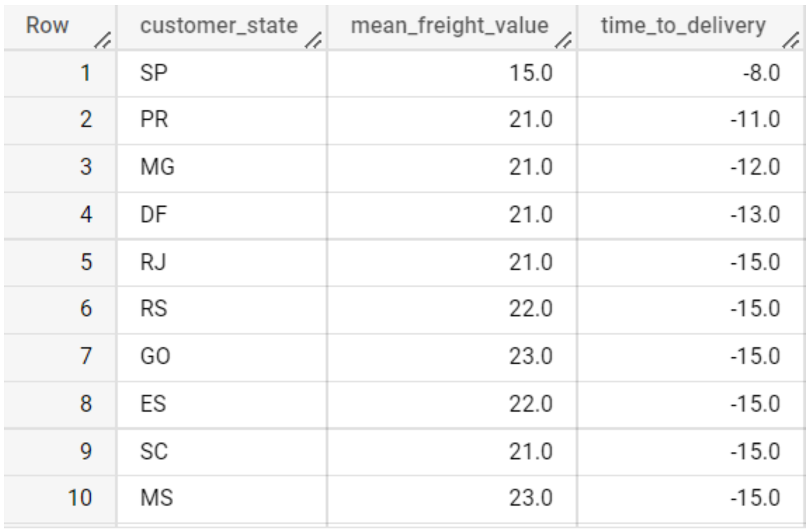
customer_state	mean_freight_value	time_to_delivery	diff_estimated_delivery
PB	43.0	-20.0	12.0
RR	43.0	-28.0	17.0
RO	41.0	-19.0	19.0
AC	40.0	-20.0	20.0
PI	39.0	-19.0	11.0

2. Top 5 states with highest/lowest average time to delivery

i. Query:

```
SELECT c.customer_state,
ROUND(AVG(i.freight_value)) AS mean_freight_value,
ROUND(AVG(DATE_DIFF(order_purchase_timestamp,
order_delivered_customer_date, DAY))) AS time_to_delivery,
ROUND(AVG(DATE_DIFF(order_estimated_delivery_date,
order_delivered_customer_date, DAY))) AS diff_estimated_delivery
FROM Target.orders o JOIN Target.customers c
ON o.customer_id = c.customer_id
JOIN Target.order_items i
ON o.order_id = i.order_id
GROUP BY c.customer_state
ORDER BY time_to_delivery DESC
```

ii. Output:



Row	customer_state	mean_freight_value	time_to_delivery
1	SP	15.0	-8.0
2	PR	21.0	-11.0
3	MG	21.0	-12.0
4	DF	21.0	-13.0
5	RJ	21.0	-15.0
6	RS	22.0	-15.0
7	GO	23.0	-15.0
8	ES	22.0	-15.0
9	SC	21.0	-15.0
10	MS	23.0	-15.0

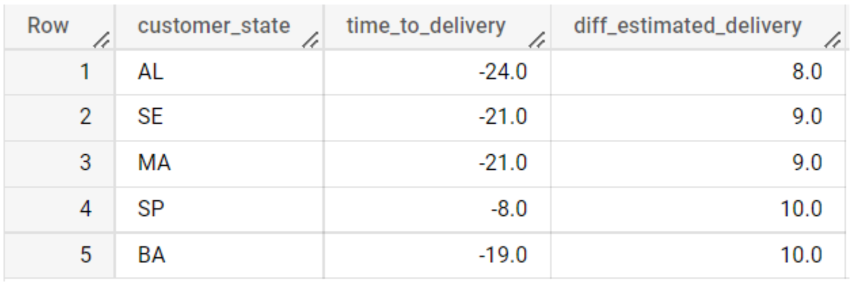
3. Top 5 states where delivery is really fast/ not so fast compared to estimated date

i. Query:

```
SELECT c.customer_state,
ROUND(AVG(DATE_DIFF(order_purchase_timestamp,
order_delivered_customer_date, DAY))) AS time_to_delivery,
ROUND(AVG(DATE_DIFF(order_estimated_delivery_date,
order_delivered_customer_date, DAY))) AS diff_estimated_delivery
FROM Target.orders o JOIN Target.customers c
ON o.customer_id = c.customer_id
JOIN Target.order_items i
ON o.order_id = i.order_id
```

```
GROUP BY c.customer_state
ORDER BY diff_estimated_delivery ASC
LIMIT 5
```

ii. Output:



Row	customer_state	time_to_delivery	diff_estimated_delivery
1	AL	-24.0	8.0
2	SE	-21.0	9.0
3	MA	-21.0	9.0
4	SP	-8.0	10.0
5	BA	-19.0	10.0

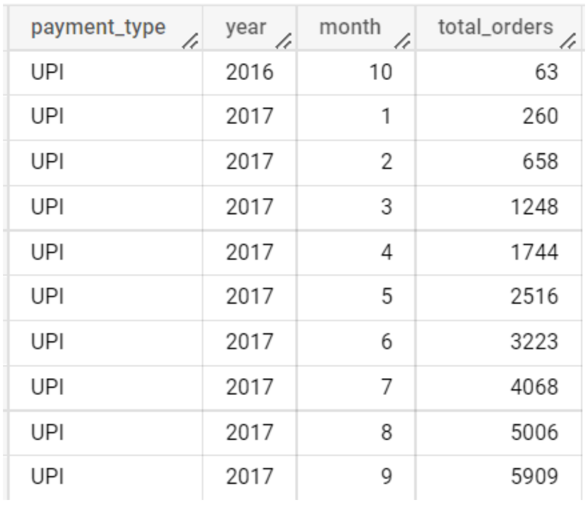
6. Payment type analysis:

1. Month over Month count of orders for different payment types

   i. Query:

```
SELECT DISTINCT p.payment_type,
EXTRACT(YEAR FROM order_purchase_timestamp) AS year,
EXTRACT(MONTH FROM order_purchase_timestamp) AS month,
COUNT(p.order_id) OVER (PARTITION BY p.payment_type
ORDER BY EXTRACT(YEAR FROM order_purchase_timestamp) ASC,
EXTRACT(MONTH FROM order_purchase_timestamp) ASC) AS total_orders
FROM Target.orders o JOIN Target.payments p
ON o.order_id = p.order_id
```

   ii. Output:



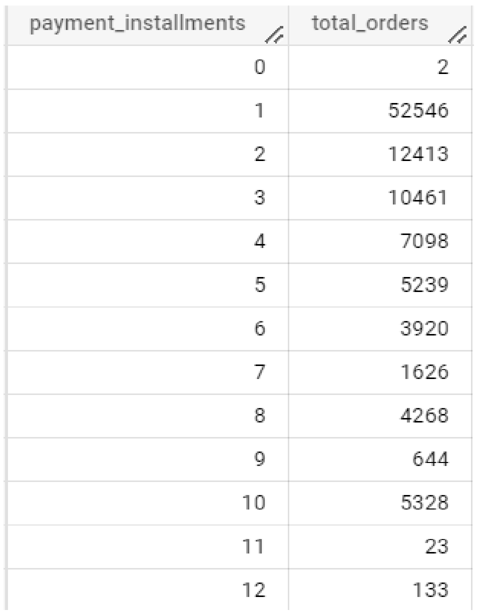
payment_type	year	month	total_orders
UPI	2016	10	63
UPI	2017	1	260
UPI	2017	2	658
UPI	2017	3	1248
UPI	2017	4	1744
UPI	2017	5	2516
UPI	2017	6	3223
UPI	2017	7	4068
UPI	2017	8	5006
UPI	2017	9	5909

2. Distribution of payment installments and count of orders

   i. Query:

```
SELECT payment_installments,
COUNT(order_id) AS total_orders
FROM Target.payments
GROUP BY payment_installments
```

ii. Output:



payment_installments	total_orders
0	2
1	52546
2	12413
3	10461
4	7098
5	5239
6	3920
7	1626
8	4268
9	644
10	5328
11	23
12	133